REMARKS

Claims 52-90 are still pending in this application. Reconsideration of the application is earnestly requested.

Applicant respectfully submits that the above amendments to the specification address the Examiner's objections regarding the typesetting errors, the embedded hyperlink and the use of trademarks. Regarding the provisional same invention double patenting rejection, Applicant submits that amendments to the claims overcome that rejection.

The Examiner has rejected claims 52-90 under 35 USC §103 as being unpatentable over *Moore et al.* and *Ernst*. Although the Examiner's arguments have been carefully considered, Applicant respectfully traverses these rejections as explained below.

The Present Invention

The independent claims have been amended to clarify the approach to the complex and highly demanding problem of autonomically implementing and verifying business logic within a heterogeneous distributed computing environment. The key problem to be solved is one of behavioral preservation across the environment, while driving the autonomic feedback loop in an effective manner.

Behavioral preservation is important because as users make changes to business logic representations those changes must be reflected across the distributed computing environment without impacting existing tasks being performed. Avoiding impacting other tasks is especially important when one considers legacy systems.

A technique of preparing executables for execution involves the steps of: software applications are described using process algebra, executable code for the applications is generated and deployed, the executable code is tested and deployed in the distributed environment, the performance of the deployed executables is monitored, and finally the executables are subsequently modified in accordance with the results of the monitoring.

The invention specifically incorporates environmental constraints specified within the process representations that are used to indicate what resources it requires from a run-time environment in order to execute correctly. Furthermore, the invention also uses contextual information to determine the most appropriate execution representation for the business logic as a key part of the autonomic feedback loop. These features are explicitly included in all of independent claims in the application and are neither taught or suggested by the prior art either alone or in combination.

In the present invention, two important steps are taken to help achieve behavioral preservation, namely:

1. The process calculus notation of the process representations is embellished with **environmental constraints** associated with the runtime environment (see pages 8 and 11); and,

2. **Contextual information** (a term defined on page 1 and described further throughout) is applied to tune the performance of the executable forms of the process representations.

The independent claims have been amended accordingly. Specifically, the claims now require that **the process representations specify one or more environmental constraints** and that the executables are altered autonomically in accordance with analyzed process execution information **and contextual information**. As explained at page 6 of the specification (for example) an *autonomic* computing system seeks to control key functionality without conscious awareness or involvement. Thus, the need for a conscious human analyst is minimized.

The Cited Art Distinguished

The prior art fails to describe a complete and practical solution to the behavioral preservation problem across a distributed computing environment while at the same time driving an automatic feedback loop in an effective manner.

The Office Action alleges that *Moore* (in paragraphs [0030] and [0031] discloses processes in a "process calculus notation" (which Applicant denies that *Moore* discloses) and that *Ernst* provides a form of optimization feedback loop.

Even though *Ernst* has identified such a feedback loop, the present invention differs because of its use of a process calculus in combination with such a feedback loop. Claims 1, 14 and 27 all require "a process calculus notation" in a first step or first element. *Ernst* discloses no such process calculus notation. Process calculus provides a more expansive description of an executable business process because it can deal with mobility; the ability to deal with mobility is important to many business processes that manage a delegatory model in which a channel or port is passed from one process instance to another, thus enabling the recipient to do something it could not do before. One is able to model classic delegation in this way.

As to the allegation in the Office Action that *Moore* uses process calculus, there is no such reference in that publication. The only XML representation that is mentioned is BRML. Indeed, *Moore* discloses describing business processes using business rules, but these are in all cases first order logic and not process algebraic. They have no expression of mobility nor do they express concurrency. They do express first order logic and can be chained together, but not in a peer-process fashion. Rather, they are implemented as a specialized rule execution algorithm that is optimized for parallel threads in a process. Where the present invention differs over *Moore* is in the use of process calculus to describe processes and all that follows should be seen in this context.

Claims 1, 14 and 27 all require use of process algebra (*i.e.*, a process calculus notation) that specifically deals with distribution by describing interactions (*e.g.* when P is composed with Q it describes how P and Q interact through message passing). If this were to be done using the techniques of *Moore*, the semantics of the message exchange would not be recorded and so causality between a rule firing in one engine because of something that occurred in another engine, regardless of the algorithm used, would be lost and therefore cannot be reasoned about. Therefore *Moore* has lost any ability to ensure behavioral typing at any level.

As a result, the level of validation the present invention provides and the correctness that we can expect is significantly higher and more profound in a distributed setting than either *Moore* or *Ernst* can achieve (either alone or in combination). Both *Moore* and *Ernst* are very much based on a process-centric and non-distributed architecture (hence the term "rule engine" used in *Moore*).

Rejection under 35 USC §101

The Office Action has rejected claims 78-90 in that the bodies of these claims only recite software elements and no hardware elements. Applicant submits that it is not required under the law that a computer system claim must recite hardware elements. Even though the modules of claim 78, for example, are described as software modules, these modules may be embodied in a computer readable medium and thus are statutory.

Reconsideration of this application and issuance of a Notice of Allowance at an early date are respectfully requested. If the Examiner believes a telephone conference would in any way expedite prosecution, please do not hesitate to telephone the undersigned at (612) 252-3330.

Respectfully submitted,
BEYER WEAVER & THOMAS

Jonathan O. Scott
Registration No. 39,364

BEYER WEAVER & THOMAS, LLP
P.O. Box 778
Berkeley, CA 94704-0778

Telephone: (612) 252-3330
Facsimile: (612) 825-6304